BTN Wallet

Smart Contract Audit

Document Info

Project Name	BTN Wallet
Project Period	01.05.2024 - 01.07.2024
Test Result	Passed
Report Prepared By	Zhang Yatao
Changelog	01.07.2024 - Release Report

Project Background

Blue Higgs is a leading research and consulting firm specializing in the web3 and crypto industries. We offer a comprehensive suite of services to help our clients navigate the complex and rapidly evolving world of decentralized finance and blockchain technology.

Our team, with years of security expertise and experience working with Fortune 500 companies and emerging technology startups, has a heavy focus on East Asia. This enables us to consistently deliver tailored recommendations and valuable insights to CISOs, CTOs, and CEOs across various industries. Our approach involves understanding the hacking risks associated with multiple sectors, including Fintech, Healthcare, Telco, Crypto, SaaS, and more. Our team members hold recognized technical certifications like CISA, CISSP, OSCP, OSCE, and AWS Certified Security.

We further nurture our security expertise through Elessar Labs, our Research labs focused on Blockchain Security. Elessar Labs has contributed to several Web3 projects including DigiFT, BitMake, Blue Helix Chain, Matr1x.io, Avatar Wallet, 996Fubao NFT, t.xyz, and more.

In this project, Blue Higgs Research Limited (the Consultant) is contracted by B S C Chain (the Customer) to conduct Smart Contract Audit . This report presents the findings of the security assessment conducted on the BTN Wallet Smart Contract.

Test Methodology and Summary

BTN Wallet is a secure and user-friendly digital wallet designed to store and manage various cryptocurrencies and digital assets. It allows users to securely access their funds anytime, anywhere without worrying about their private key. BTN Wallet supports a wide range of cryptocurrencies and blockchains, including any EVM compatible chains and tokens.This project is basically forked from other mature wallet projects, just doing the basic changes related to the verification of special account privileges, so the overall risk is relatively low.

BTN wallet's trust in the user is constructed on top of the OpenID protocol .Simply put, where a user needs to manipulate a wallet, he first needs to sign the information through a third party
OpenID service (e.g. google or facebook, etc.). The signature is then sent to a smart contract for verification. The smart contract will verify the third party's public key and signature information. The key to security lies in protecting the integrity of the public key and ensuring the integrity of the command validation process.

Our audit scope is as below

Project Repo	https://*PRIVATEREPO*/btnwallet/wallet-contract
Audit Commit	3a6ea857a4eea6094ec66d96489e1a270c0d288e

- **Manager.sol**
- **OpenID.sol**
- **SimpleProxy.sol**
- **SingletonFactory.sol**
- **Wallet.sol**
  - **IERC223Receiver.sol**
  - **IManager.sol**
  - **IModuleAccount.sol**
  - **IModuleHooks.sol**
  - **IOpenID.sol**
- **modules**
  - **ModuleGuest.sol**
  - **ModuleMain.sol**
  - **commons**
    - **Implementation.sol**
    - **ModuleAccount.sol**
    - **ModuleAuth.sol**
    - **ModuleBase.sol**
    - **ModuleCall.sol**
    - **ModuleHooks.sol**
    - **ModuleStorage.sol**
    - **ModuleTransaction.sol**
  - **utils**
    - **FeeEstimator.sol**
    - **GasEstimator.sol**
    - **LibEmailHash.sol**
    - **LibTimeLock.sol**
    - **LibUnipassSig.sol**
- **utils**
  - **LibBase64.sol**
  - **LibBytes.sol**
  - **LibModexpPrecompile.sol**
  - **LibOptim.sol**
  - **LibRole.sol**
  - **LibRsa.sol**
  - **LibSignatureValidator.sol**

After our inspection and auditing according to industry standards, we found NO CRITICAL, HIGH risk problems. There are 2 LOW risk problems and several notes on best practice. Please refer to Audit Details.

Audit Detail

ID:	R1	File:	contracts/Manager.sol	Risk:	Low
Finding:	Lacking control measures on privileged roles				
In contracts/Manager.sol, there are several special privilege roles. These include special roles for upgrading contracts and updating OpenID validation keys. When performing privileged operations and changing address permissions, the amendment will be effective immediately. We suggest that involved controls like Time Lock to control the risk of special roles' private key leakage					

ID:	R2	File:	contracts/OpenID.sol#_validateTimestamp	Risk:	Low
Finding:	Bypass Timestamp validation in certain chain Id.				
In contracts/OpenID.sol#_validateTimestamp, there is a if statement used to bypass the check of iat and exp when the chain id is equal to 31337. ***Update*** *: team response: chain id 31337 is hardhat testnet, prefer to maintain everything in one place .*					

ID:	N1	File:	N/A	Advices
Finding:	Note the risk of Phishing			
As the user's authentication page will be provided by a third party(OpenID Service provider), it is recommended that the user be reminded of the need to pay attention to whether or not he or she is signing a legitimate request when using the wallet.				

ID:	N2	File:	contracts/Manager.sol#_authorizeUpgrade	Advices
Finding:	Suggest allowUpgrade need to check whether upgradeTo has been included			
In contracts/Manager.sol#_authorizeUpgrade, when upgrading the implementation to a newer version, the contract didn't check whether the upgradeTo method exists in the new implementation. In that case, the new contract cannot be upgraded again.				

ID:	N3	File:	contracts/modules/utils/*	Advices
Finding:	Existence of redundancy code			
This project is forked from other mature contracts and simplified, so some useless code is not removed. Such as contracts/modules/utils/LibEmailHash.sol, it is used to calculate the hash of an email address. The contracts in contracts/modules/utils/ are mostly unused.				

ID:	N4	File:	N/A	Advices
Finding:	Some CEXs may reject the withdrawal to a smart contract wallet.			
In CEX, when withdrawing digital currency, if the gas consumption is higher than the CEX setting, the withdrawal will be rejected. Since ERC20 token transfers to contract wallets consume slightly more gas than transfers to EOA addresses, CEX withdrawals will be denied.				

ID:	N5	File:	contracts/Manager.sol#getImplementationRole	Advices
Finding:	Note the allowImplementation() and allowHook() just allow one address			
In contracts/Manager.sol#getImplementationRole, it looks like there can be a storage slot to store the address of the implementation, need to confirm that the logic is in line with expectations. Since _implementationCheck is checked when the user invokes updateImplementation, logically it should allow many different implementations to exist at the same time. allowHook() has a similar situation.				

ID:	N6	File:	contracts/modules/commons/ModuleAuth.sol#constructo	Advices
Finding:	Irrelevant revert message			
In contracts/modules/commons/ModuleAuth.sol#constructor(), there is a check that the implementation of open ID is not zero address. But the revert message is irrelevant ***Update*** *: Fixed*				

Disclaimer

ElessarLabs issues this report only based on the facts that have happened or existed before the report is issued, and will take the corresponding responsibilities for the report based on these facts. Regarding any unknown vulnerabilities or security incidents that happen or exist after the issue of this report, ElessarLabs cannot verify their security conditions and will not be responsible for them. All of the security audits analysis and other contents in this report are only based on the files and documents provided to ElessarLabs by information providers(hereinafter referred to as "provided documents"). ElessarLabs assumes that the provided documents are not under any of these circumstances, such as being absent, being tampered, being abridged or being concealed. If the information of the provided documents were absent, tampered, abridged, concealed, or did not conform to the reality, ElessarLabs would not be responsible for any of the loss or disadvantages caused by these circumstances. ElessarLabs only performs the appointed security audits for the security condition of this project and issues this report. ElessarLabs is not responsible for the background of this project or any other circumstances.